Inference-Time Prior Adaptation for Imbalanced Data Streams under Concept Drift

Tobias Callies[1](✉), Eirini Ntoutsi[1], and Arthur Zimek[2]

[1] University of the Bundeswehr Munich, Germany; tobias.callies@unibw.de
[2] University of Southern Denmark, Odense, Denmark

Abstract. Data stream classification under concept drift and class imbalance requires models to adapt continuously while maintaining strong minority-class performance. Many existing approaches rely on *active memory management*, including instance removal, memory restructuring, resampling, and synthetic data generation. We present PAI-KNN, a k-nearest neighbor classifier that adapts via inference-time prior adjustment rather than memory modification. PAI-KNN maintains a simple sliding-window buffer and dynamically adjusts its decision rule using class prior estimates derived from recent observations. Evaluated on a multitude of synthetic data streams covering various imbalance and drift scenarios, PAI-KNN achieves competitive performance with state-of-the-art methods, performing particularly well under scenarios involving dynamic class imbalance and concept drift. These findings highlight the potential of inference-time adaptation in challenging non-stationary environments and suggest that adaptation in memory-based stream classifiers need not rely exclusively on memory modification.

Keywords: data stream classification · class imbalance · concept drift · inference-time adaptation · prior adaptation · k-nearest neighbors

1 Introduction

Data stream classification [17] often requires models to operate under two simultaneous challenges: *concept drift* and *class imbalance*. As data distributions evolve over time, classifiers must continuously adapt to changing data while maintaining reliable performance on underrepresented classes. This problem arises in numerous real-world applications, including fraud detection, network monitoring, and predictive maintenance.

Existing approaches address concept drift and class imbalance in different ways. Many stream classifiers are *model-based*, maintaining either a single evolving model [10], [18] or a pool of models that are continuously updated, reweighted, or replaced over time [5], [12], [8], [19], often combined with drift detectors like ADWIN [4]. K-nearest neighbor (KNN) classifiers, in contrast, are *instance-based or lazy learners* that maintain explicit memories of previously observed instances and perform prediction directly from stored observations. Several KNN-based

stream classifiers have been proposed that address concept drift and class imbalance through increasingly sophisticated *active memory management mechanisms* [13, 1]. Adaptation is typically achieved through instance removal, restructuring, compression, or balancing operations. While effective, active buffer-management strategies increase algorithmic complexity and may prematurely alter or discard useful information.

In this paper, we investigate whether adaptation in KNN-based stream classifiers must occur through memory modification. We propose PAI-KNN, a k-nearest neighbor classifier that maintains a simple sliding-window memory and adapts through inference-time prior adjustment. By shifting adaptation from memory management to inference, PAI-KNN responds to changing class distributions without the need for expensive memory management mechanisms. We evaluate PAI-KNN on a benchmark of synthetic data streams [2] covering diverse imbalance and drift scenarios. Despite its simplicity, PAI-KNN achieves competitive performance with state-of-the-art methods and performs particularly well under scenarios involving high class imbalance and concept drift. These findings suggest that inference-time prior adaptation can provide a simple and effective alternative to complex memory management by avoiding structural assumptions.

The rest of the paper is organized as follows: Section 2 introduces the preliminaries and basic concepts. Related work is discussed in Section 3. In Section 4, we present our proposed method PAI-KNN. The experimental evaluation is detailed in Section 5. Conclusions and outlook are provided in Section 6.

2 Preliminaries and basic concepts

A data stream is a potentially infinite sequence of instances $D = (x_1, \ldots, x_t, \ldots)$ where $x_t \in \mathbb{R}^d$ denotes a d-dimensional feature vector arriving at time t. Under the standard prequential evaluation protocol, an observation x_t is first classified, producing a prediction $\hat{y}_t$. The true label y_t is revealed subsequently and can be used to update the classifier. Instance-label pairs (x_t, y_t) are assumed to be sampled from evolving data distributions $\mathbb{P}_t(x, y)$. Concept drift [7] may hence arise from *class priors* ($\mathbb{P}_{t+1}(y) \neq \mathbb{P}_t(y)$), *likelihoods* ($\mathbb{P}_{t+1}(x|y) \neq \mathbb{P}_t(x|y)$), or *posteriors* ($\mathbb{P}_{t+1}(y|x) \neq \mathbb{P}_t(y|x)$) changing.

For classification problems, the *imbalance ratio IR* quantifies the class-prior distribution and is defined as the (potentially infinite) ratio of majority to minority class frequencies over a set of instance labels (in case of multi-class problems, ratio of largest to smallest class). Consequently, changes in class priors are directly reflected in changes in the imbalance ratio. Since PAI-KNN adapts through dynamic prior adjustment, such changes are of particular interest in this work. In many real-world streams, class frequencies do not only exhibit strong imbalance but may also change over time, resulting in *dynamic imbalance ratios*. Depending on the speed of change, such shifts may occur either *suddenly* or *gradually*. Due to the potentially infinite nature of data streams and processing speed requirements, memory-based classifiers typically maintain only a bounded subset of previously observed instances (also known as *buffer*).

3 Related work

We divide related work into three parts: classifiers for imbalanced data streams (Section 3.1), KNN classifiers for data streams (Section 3.2) and KNN classifiers for imbalanced static data (Section 3.3).

3.1 Imbalanced stream classifiers

Tree-based methods and their ensembles remain the dominant approach to stream classification. For imbalanced streams, numerous methods have been proposed that combine drift detection and adaptation with resampling, cost-sensitive learning, or imbalance-aware ensemble construction [2]. For the purpose of this work, the distinction between inference-based interventions (adjustments to the prediction rule) and data-based interventions (modifications of the data presented to the learner) is particularly important.

We therefore restrict our comparison for such methods to two representative state-of-the-art methods selected due to their performance [2]: *Cost-sensitive Adaptive Random Forest* (CSARF) [12] introduces a cost matrix to tackle imbalanced performance using adaptive random forests at inference time. *Synthetic Minority Oversampling Technique with Online Bagging* (SMOTE-OB) [3] is the best-performing method among methods heavily influenced by the SMOTE [6] algorithm, generating synthetic data on subsets of features, which are then used to continually update an ensemble of tree learners.

3.2 KNN-based stream classifiers

Self-Adjusting Memory KNN (SAM-KNN) [13] addresses concept drift through active memory management. The method maintains a short-term memory (STM) containing recent observations and a long-term memory (LTM) storing compressed historical information. As the stream evolves, SAM-KNN continuously restructures these memories based on the label consistency of stored instances with recent observations. Following drift detection, instances that remain consistent with the STM are retained, whereas inconsistent instances may be permanently removed. Adaptation is therefore achieved through active modifications of the stored memory.

Drift-Aware Multi-Memory Model (DAM3) [1] extends SAM-KNN to handle class-imbalanced streams. To improve minority-class performance, DAM3 introduces a third Working Memory (WM) and explicitly maintains a balanced class representation while identifying and removing class inconsistencies in the stored data. As in SAM-KNN, adaptation is achieved through active management of the underlying memories.

Both approaches achieve adaptation through active buffer management, i.e., modifications of the stored memories. SAM-KNN and DAM3 continuously inspect, restructure, and curate stored instances to maintain memory consistency and adapt to changing stream characteristics. While effective, such mechanisms

require ongoing memory interventions and may permanently alter or discard information based on potentially uncertain evidence of change. This is particularly consequential in evolving streams, where instances initially regarded as outliers or noise may later prove representative of a changing class distribution or decision boundary [14]. Prematurely discarding them may therefore remove valuable evidence of concept evolution.

3.3 Static inference-time prior-adjusted KNN

An inference-time prior-adjusted KNN for static imbalanced learning was proposed in [9]. The method adjusts the decision rule rather than the data to compensate for class imbalance and achieves performance comparable to data-level interventions such as SMOTE [6] at a substantially lower computational cost.

In this formulation, KNN classification is viewed as a maximum-a-posteriori (MAP) estimator, where the class priors $\mathbb{P}(y_i) = N_i/N$ are estimated from the data and the likelihood is approximated by $\mathbb{P}(x|y_i) = \frac{n_i}{N_i} V(x)$. Here n_i is the number of instances of class y_i among the k nearest neighbors of an instance x, N_i is the total number of instances of class y_i in the dataset, N is the total number of instances, m will denote the number of classes, and $V(x)$ is the volume of the neighborhood.

Under mild assumptions, and using Bayes' theorem, the posterior can then be written as:

$$\mathbb{P}(y_i \mid x) \propto \frac{\frac{n_i}{N_i} V(x) \frac{N_i}{N}}{\sum_{j=1}^{m} \frac{n_j}{N_j} V(x) \frac{N_j}{N}} = \frac{n_i}{\sum_{j=1}^{m} n_j} = \frac{n_i}{k}$$

To improve performance on underrepresented classes, the prior estimate is substituted by a uniform prior: $\mathbb{P}(y_i) = \frac{1}{m}$. The new decision rule then becomes: $\hat{y} = \arg\max_{\{j=1,\dots,m\}} \frac{n_j}{N_j}$.

In this work, we extend such inference-time prior adjustment from static imbalanced learning to non-stationary data streams with evolving class priors.

4 PAI-KNN: Inference-Time Prior Adaptation KNN

Motivated by the promising results of the prior-adjusted static KNN [9] for static imbalance learning and by the prevalence of increasingly sophisticated active memory management mechanisms in KNN-based stream classifiers, we investigate whether adaptation can be achieved through inference-time prior adjustment rather than active memory management. To this end, we extend the concept of inference-time prior adjustment to non-stationary data streams.

An overview of the proposed approach is shown in Fig. 1. Relative to a vanilla KNN classifier, PAI-KNN augments prediction with a prior-adjustment mechanism that adapts the decision rule to changing class distributions over time. Section 4.1 introduces the model components, followed by the update and inference steps in Section 4.2. Computational complexity is discussed in Section 4.3.

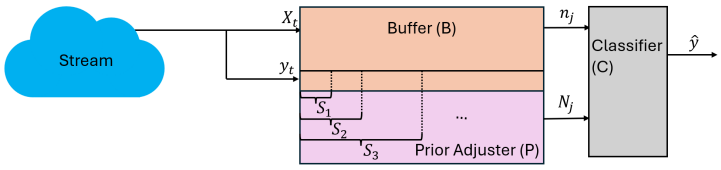

Fig. 1. Model components (B),(P),(C). Target labels are received after prediction.

4.1 Model definition

PAI-KNN augments a standard sliding-window KNN classifier with a prior-adjustment module that estimates class priors from recent observations and incorporates them into the prediction rule. Concretely, PAI-KNN consists of three components explained hereafter:

- a **buffer** B of size N
- a **prior-adjustment module** P, which estimates class priors from recent stream history
- a **KNN classifier** C, which combines neighborhood information with the estimated priors during inference.

Buffer B**:** The buffer B is implemented as a FIFO queue and stores recent observations from the stream.

Prior-adjuster P**:** The prior-adjustment module P estimates the current class prior distribution from recent stream observations. To do so, PAI-KNN maintains multiple sliding windows of different sizes. The multiple windows allow the classifier to adaptively select the longest statistically consistent history for prior estimation, thereby limiting responsiveness to statistically significant changes. Class prior estimates are then obtained from the label frequencies within the selected window and incorporated into the prediction rule.

Concretely, we use L sliding windows $S_1, \ldots, S_L$ that store the latest $2^{l_{\min}}, \cdots, 2^{l_{\max}}$ labels, which can be effectively implemented using the existing buffer queue. The largest window is chosen to match the buffer size, i.e., $l_{\max} = \log_2(|B|)$, and $l_{\min} \geq 5$ is recommended for efficient hypothesis testing. At time t, the sliding windows are given by

$$S_k(t) = \{y_i \mid t - i \leq 2^{l_{\min}+k-1}, i \geq 1\}$$

The module further maintains a *significance threshold* p used to compare label distributions across windows. During inference, a χ^2 test is used to assess whether two windows are plausibly generated from the same distribution.

The KNN classifier C**:** The KNN classifier C operates on the buffer B and is associated with the typical KNN hyperparameters: the number of neighbors k and the distance function $d(\mathbb{R}^d \times \mathbb{R}^d) \to \mathbb{R}^+$. In addition, C includes a parameter $\theta \in [0, 1]$, interpolating between uniform priors and the observed class priors. This interpolation provides a direct trade-off between overall accuracy and minority-class performance, with $\theta = 1$ corresponding to classical KNN and $\theta = 0$ optimizing for recall on the minority class, as in knn-BPP [9].

4.2 Model update and inference

Update step: it only consists of queue insertions of (x, y) into the buffer B, and an insertion of the label y into the sliding windows S_j.
Inference step: For an instance x, the classifier C determines for every class $y_j \in Y$ the number of instances n_j among the k nearest neighbors of x. The prior-adjustment module P selects the longest statistically consistent history and uses the corresponding label frequencies to estimate the current class priors. In particular, P provides for every class the number of samples $N_j = |\{y = y_j \mid y \in S^*\}|$ in the longest consistent sliding window $S^* = S_{k^*}$, where

$$k^* = \min\left(\{1 \leq k < L \mid \chi^2(S_k, S_{k+1} \setminus S_k) < p\} \cup \{L\}\right)$$

With all n_j and N_j determined that way, C adjusts the prior using θ to effectively choose the sensitivity to imbalance and calculates the following posterior:

$$p(y_i \mid x) = \frac{\frac{n_i}{N_i}\left(\theta\frac{N_i}{N} + (1-\theta)\frac{1}{m}\right)}{\sum_{j=1}^{|Y|}\frac{n_j}{N_j}\left(\theta\frac{N_j}{N} + (1-\theta)\frac{1}{m}\right)}$$

The class label of x is the one that maximizes the posterior.

4.3 Computational Complexity

The time complexity of the *update step* is effectively negligible as only a queue insertion takes place. Time complexity in the *inference step* is dominated by the neighborhood search and hence worst-case complexity naively scales with $\mathcal{O}(Nd^2)$, where N is the buffer size and d is the feature space dimensionality. Compared to active buffer management methods, worst-case complexity is therefore significantly faster. For example, SAM-KNN scales slightly worse than $\mathcal{O}(N^2d^2)$ [13]. The space complexity of our method scales with $\mathcal{O}(Nd)$.

5 Experiments

We evaluate PAI-KNN on a benchmark of synthetic data streams against state-of-the-art competitors. Section 5.1 describes the evaluation design, datasets, competitors, metrics, and experimental protocol before presenting comparative results (Section 5.2) and a parameter analysis (Section 5.3) with respect to the prior-adjustment trade-off parameter θ, and the neighborhood size k.

5.1 Evaluation design

Datasets: We adopt the synthetic stream benchmark [2]. Streams consist of 200.000 instances, and are generated using multiple random generators to evaluate methods across different feature spaces. We restrict analysis to (five) binary generators producing only numerical features to avoid the need to craft distance

functions on categorical features. Benchmark scenarios fall into four categories: i) static imbalance ratio, ii) dynamic imbalance ratio, iii) instance-level difficulties, and iv) concept drift.

Competitors: We compare our approach to the competitor methods discussed in Section 3: CSARF [12], SMOTE-OB [3], and SAM-KNN [13]. For DAM3 ([1]), the available online implementations struggled with long stream sizes. For CSARF and SMOTE-OB, we use results and hyperparameters from a recent survey [2], where they are set according to their self-reported standard configuration. For SAM-KNN, experiments were performed using the reference implementation in [16]. To validly compare against our own method, buffer size was set to 1024.[3] For PAI-KNN, we chose $N = 1024$, sliding windows in the prior adjuster ranging from size 64 to 1024, $p = 0.01$ as a statistical significance threshold and $k = 20$. $\theta = 0$ was chosen, thereby prioritizing G-Mean performance.

Although the choice of distance function can have an important impact, it is orthogonal to this work. We therefore use the L_2 norm throughout to avoid confounding the effects of the prior intervention with those of the chosen metric.

Evaluation Setup: We run all experiments using River [15], with data streams generated using the code of [2]. In line with their work [2], we report the median obtained from running methods on 5 streams generated with different random seeds. Evaluation metrics for each run are calculated by taking the arithmetic mean over sliding windows of size 500. We aggregate metrics over groups of different synthetic streams, using the arithmetic mean. Code is available[4].

Evaluation Metrics: Performance is evaluated using Accuracy, Cohen's Kappa, and G-Mean. Accuracy measures overall correctness, Kappa accounts for chance agreement, and G-Mean emphasizes balanced performance across classes.

We do not report detailed computational costs as these are very platform and implementation dependent [11]. To give some reference nonetheless, our not-optimized Python implementation of PAI-KNN processes approximately 1000 instances in two seconds, depending on the dimensionality of the feature space.

5.2 Predictive performance

To better understand the behavior of PAI-KNN under changing imbalance ratios, we plot in Fig. 2 the performance for PAI-KNN, SAM-KNN, and a baseline KNN method (operating on a FIFO buffer without any prior adjustment, with $k = 20$, $N = 1024$) for a concrete scenario in which the imbalance ratio reverses twice over time (*reflipping*). We see that SAM-KNN's imbalanced performance deteriorates notably more quickly than the baseline KNN method for high imbalance ratios. On the other hand, PAI-KNN maintains a high G-Mean level throughout even significant gradual changes. We interpret this as follows: maintaining buffer consistency is detrimental if streams change significantly and rapidly enough.

[3] H-params SAM-KNN: $k = 5$, $N = 1024$, ltm_size= 0.4, min_stm_size= 50, weighting=uniform, use_ltm=True, stm_size_option=maxACCApprox

[4] https://github.com/tcallies/streamKNN

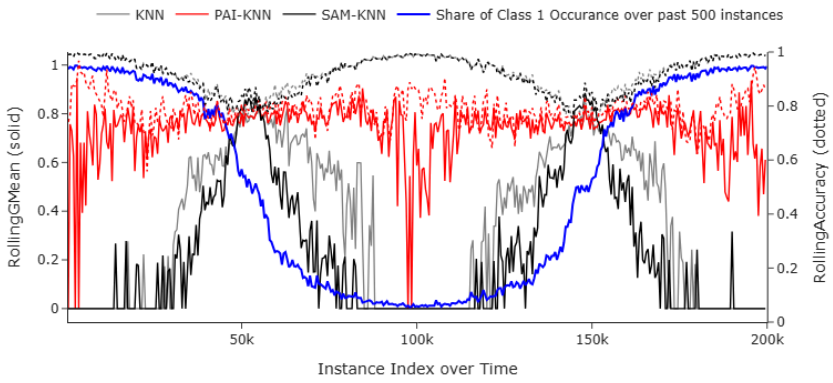

Fig. 2. G-Mean and Accuracy over time - HyperplaneGenerator, gradual reflipping

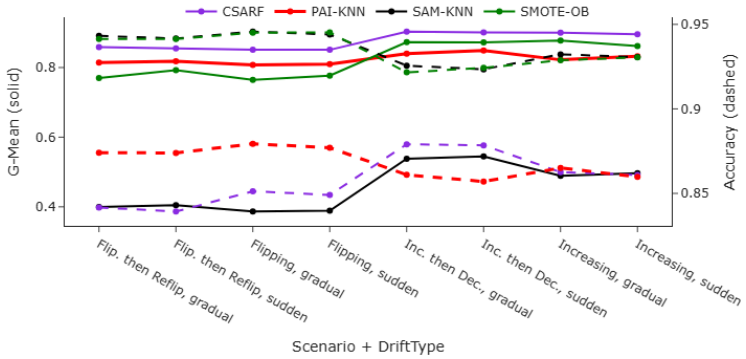

Fig. 3. G-Mean and Accuracy per method for streams with dynamic IR

This observation is reflected in the results shown in Fig. 3, where we plot G-Mean and Accuracy for four different scenarios of imbalance ratio changes, with changes occurring either gradually or suddenly, and results aggregating over streams from different generators. PAI-KNN performs on par with ensemble competitors in terms of G-Mean, while underperforming in Accuracy. Sudden changes describe an instantaneous and significant shift in the data generating distribution, which for gradual changes is prolonged through interpolation over several thousands timesteps.

CSARF, similarly to us, excels in terms of G-Mean at the sacrifice of Accuracy, showcasing the natural conflict arising from cost-sensitive adjustments.

Concept Drift We limit analysis to streams with both gradual and sudden feature space concept drift, for different, static imbalance ratios. We plot in Fig. 4 performance by drift type and imbalance ratio, and aggregated across generators. We see that in terms of G-Mean, we outperform competitors for

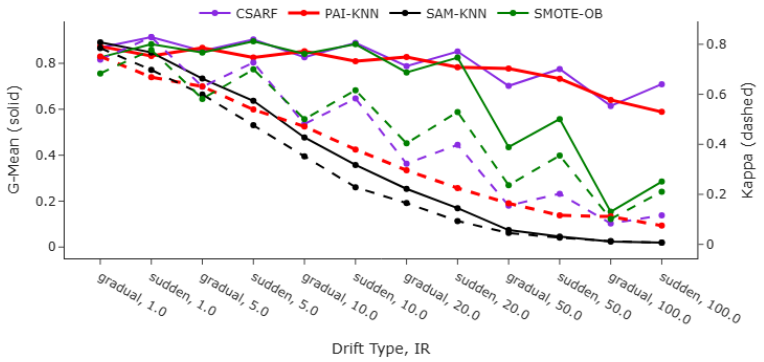

Fig. 4. G-Mean and Kappa per method for concept drift streams

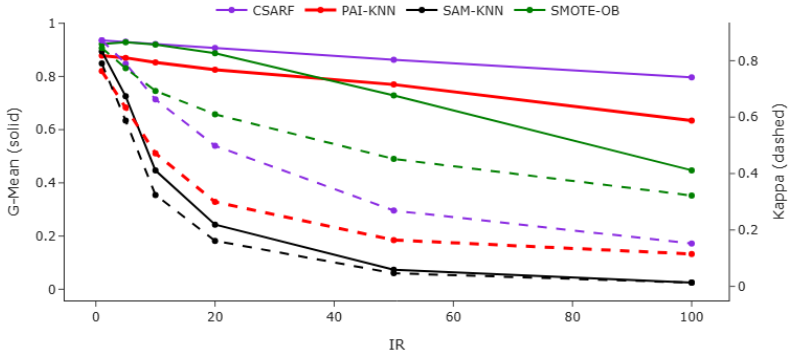

Fig. 5. G-Mean and Kappa per method for static streams with varying *IR*

gradual changes, and high imbalance ratios. Again, a very clear data-intrinsic tradeoff can be recognized that is resolved in favor of G-Mean for PAI-KNN and CSARF, whereas performance for SAM-KNN correlates more strongly with SMOTE-OB.

Static Imbalance Ratio Streams in this category have no intrinsic temporal component; both feature and label distribution remain fixed over time. Here, the ability to retain and add information over time drives performance, whereas the ability to forget—a focus point of PAI-KNN—is irrelevant. As a result, PAI-KNN performs generally worse than competitors, but robust to very high *IR* (Fig. 5).

Instance Level Difficulties Streams with instance level difficulties (ILD) incorporate one or multiple challenges to differentiate data: Imbalance ratios changes, class-specific concept drift, many instances in borderline regions or in rare locations. For those ILD streams involving concept drift we plot results in

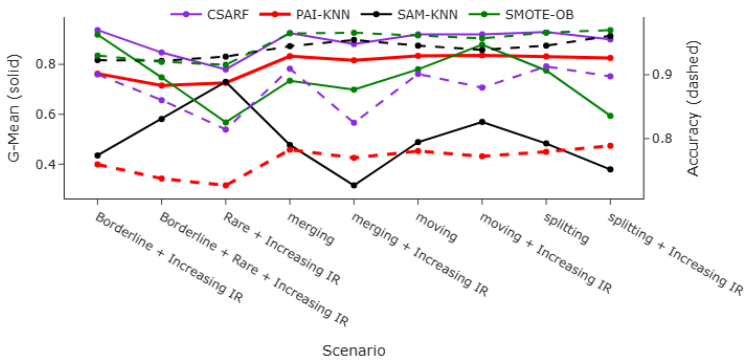

Fig. 6. G-Mean and Accuracy per method for ILD streams with concept drift.

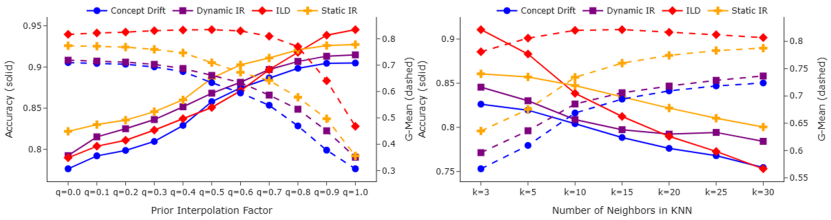

Fig. 7. Effects of θ and k per Experiment Group

Fig. 6. In terms of G-Mean, we keep up with state of the art competitors in this case despite the generator synthesizing the ILD streams producing distributions not particularly well-suited to a L_2 distance function.

We observe that SAM-KNN performs very well for the scenario involving rare instances. This is in line with our understanding that the buffer cleaning and management mechanism can perform very well when underlying assumptions are met and retaining information is important. PAI-KNN on the other hand performs more evenly across ILD scenarios: due to imposing only weak assumptions, we remain agnostic to different setups.

5.3 Parameter Effect

We evaluate parameter effects using a subset of 36 streams, grouped by scenario class.

For the *trade-off parameter* θ, $\theta = 1$ maximizes accuracy, while $\theta = 0$ optimizes G-Mean. Notably, G-Mean performance declines only slightly up to $\theta = 0.4$, offering a beneficial tradeoff. For streams exhibiting instance level difficulties (ILD), G-Mean even slightly improves up to $\theta = 0.5$, which showcases that under the right conditions, significant accuracy gains can be achieved using PAI-KNN without sacrificing on imbalanced performance. For the *neighborhood size* k, we again fix $\theta = 0$. Low values of k show higher accuracy while G-Mean

benefits from larger k. Again, the optimal tradeoff is both dependent on performance preferences and stream characteristics.

6 Conclusion

We presented PAI-KNN, a KNN-based stream classifier that introduces inference-time adaptation for jointly addressing concept drift and class imbalance. Unlike existing approaches that adapt primarily by modifying stored instances, PAI-KNN maintains a simple sliding-window memory and adapts predictions through dynamic class-prior adjustment. Our experimental evaluation demonstrated that PAI-KNN achieves competitive performance across a wide range of imbalance and drift scenarios, while performing particularly well under dynamic class imbalance and concept drift. These results indicate that effective adaptation in KNN-based stream classification does not require continuous expensive memory operations. Instead, shifting adaptation to inference offers a simple and responsive mechanism for handling rapidly changing stream conditions.

More broadly, our findings suggest that adaptation in memory-based stream classifiers need not rely exclusively on increasingly sophisticated modifications to stored instances. The simplicity of PAI-KNN allows it to react quickly in highly dynamic settings without permanently altering the stored evidence. Actively managed memories, by contrast, can improve information retention over longer horizons when the assumptions underlying their management mechanisms hold, but may prematurely modify or discard instances based on uncertain evidence of change. Memory-based and inference-time adaptations should therefore be viewed as complementary design choices, with memory adaptations offering benefits for information preservation over long horizons and inference adaptations better equipped to tackle dynamic imbalance ratio shifts.

Future work will extend inference-time adaptation beyond class-prior adjustment by exploring feature-aware drift signals, and dynamic neighborhood selection, as well as its integration with memory-based adaptation.

References

1. Abolfazli, A., Ntoutsi, E.: Drift-Aware Multi-Memory Model for Imbalanced Data Streams. In: 2020 IEEE International Conference on Big Data (Big Data). pp. 878–885 (Dec 2020)
2. Aguiar, G., Krawczyk, B., Cano, A.: A survey on learning from imbalanced data streams: taxonomy, challenges, empirical study, and reproducible experimental framework. Machine Learning (Jun 2023)
3. Bernardo, A., Valle, E.D.: SMOTE-OB: Combining SMOTE and Online Bagging for Continuous Rebalancing of Evolving Data Streams. In: 2021 IEEE International Conference on Big Data (Big Data). pp. 5033–5042 (Dec 2021)
4. Bifet, A., Gavaldà, R.: Learning from Time-Changing Data with Adaptive Windowing. In: Proceedings of the 2007 SIAM International Conference on Data Mining. pp. 443–448. Society for Industrial and Applied Mathematics (Apr 2007)

5. Cano, A., Krawczyk, B.: ROSE: robust online self-adjusting ensemble for continual learning on imbalanced drifting data streams. Machine Learning **111**(7), 2561–2599 (Jul 2022)
6. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research **16**, 321–357 (Jun 2002)
7. Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., Bouchachia, A.: A survey on concept drift adaptation. ACM Computing Surveys **46**(4), 1–37 (Apr 2014)
8. Gomes, H.M., Bifet, A., Read, J., Barddal, J.P., Enembreck, F., Pfharinger, B., Holmes, G., Abdessalem, T.: Adaptive random forests for evolving data stream classification. Machine Learning **106**(9-10), 1469–1495 (Oct 2017)
9. Gøttcke, J.M.N., Zimek, A.: Handling class imbalance in k-nearest neighbor classification by balancing prior probabilities. In: Similarity Search and Applications - 14th International Conference, SISAP 2021, Dortmund, Germany, September 29 - October 1, 2021, Proceedings. pp. 247–261 (2021)
10. Hulten, G., Spencer, L., Domingos, P.: Mining time-changing data streams. In: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 97–106 (2001)
11. Kriegel, H., Schubert, E., Zimek, A.: The (black) art of runtime evaluation: Are we comparing algorithms or implementations? Knowl. Inf. Syst. **52**(2), 341–378 (2017)
12. Loezer, L., Enembreck, F., Barddal, J.P., De Souza Britto, A.: Cost-sensitive learning for imbalanced data streams. In: Proceedings of the 35th Annual ACM Symposium on Applied Computing. pp. 498–504. ACM, Brno Czech Republic (Mar 2020)
13. Losing, V., Hammer, B., Wersing, H.: KNN Classifier with Self Adjusting Memory for Heterogeneous Concept Drift. In: 2016 IEEE 16th International Conference on Data Mining (ICDM). pp. 291–300. IEEE, Barcelona, Spain (Dec 2016)
14. Masud, M., Gao, J., Khan, L., Han, J., Thuraisingham, B.M.: Classification and novel class detection in concept-drifting data streams under time constraints. IEEE Transactions on Knowledge and Data Engineering **23**(6), 859–874 (2011). https://doi.org/10.1109/TKDE.2010.61
15. Montiel, J., Halford, M., Mastelini, S.M., Bolmier, G., Sourty, R., Vaysse, R., Zouitine, A., Gomes, H.M., Read, J., Abdessalem, T., et al.: River: machine learning for streaming data in python (2021)
16. Montiel, J., Read, J., Bifet, A., Abdessalem, T.: Scikit-multiflow: A multi-output streaming framework. Journal of Machine Learning Research **19**(72), 1–5 (2018)
17. Read, J., Zliobaite, I.: Supervised learning from data streams: an overview and update. ACM Computing Surveys **57**(12), 1–31 (2025)
18. Wagner, S., Zimmermann, M., Ntoutsi, E., Spiliopoulou, M.: Ageing-based multinomial naive bayes classifiers over opinionated data streams. In: Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2015, Porto, Portugal, September 7-11, 2015, Proceedings, Part I 15. pp. 401–416. Springer (2015)
19. Wang, S., Minku, L.L., Yao, X.: Dealing with multiple classes in online class imbalance learning. In: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence. pp. 2118–2124. IJCAI'16, AAAI Press, New York, New York, USA (Jul 2016)